過去のPBLの情報を用いた工数見積り支援ツールの開発

齋藤 尊　新美 礼彦　伊藤 恵

企業におけるソフトウェア開発では，FP 法や COCOMO などの工数見積り手法が使われている．これらの手法では，機能の複雑さやエンジニアの技術などの補正係数を利用する．しかし，ソフトウェア開発 PBL(Project-Based Learning) では，開発経験の浅い PBL 参加者がそれらを算出するのは難しいため，これらの手法を PBL で活用するのは困難である．そのため，PBL を対象とした工数の見積り方法が求められている．一方 PBL では，そのプロジェクト活動中に様々なデータを蓄積することができる．しかし，既に終了したプロジェクトも含め，それらのデータが十分に活用されていない．そこで本研究では，過去および進行中の PBL で蓄積されたデータを利用して，各成果物作成の工数見積りを行うツールを提案する．本ツールを進行中の PBL に適用することで，PBL に参加している学生たちが正確に作業見積りができるようになることが期待される．

Man-hour estimate technique such as Function Point method or COCOMO(constructive cost model) is generally used in software development companies. It need some correction coefficients. But, in software development PBL(Project-Based Learning), it is difficult that PBL participants who do not have so much software development experience use these estimate technique because they may not have enough skills for calculating these coefficients. On the other hand, we can accumulate various data in PBL. However, these data have not been inflected fully. So, in this paper, we propose a man-hour estimation tool for making deliverables which uses some data accumulated from already ended and on-going PBL. It is expected that if on-going PBL participants use this tool, they will become to be able to estimate a workload correctly.

1　はじめに

近年，ソフトウェア開発教育において，Project-Based Learning(PBL) と呼ばれる実践型教育手法が注目を集めている．PBL は通常の講義形式の教育ではなく，学習者同士が数人から十数人のグループを作り，プロジェクトを通して問題発見・解決を経験させる教育法である．特にソフトウェア開発 PBL の場合，コーディングやテストなどの下流工程だけでなく，要件定義や内部設計などのソフトウェア開発の上流工程を経験することで，学習者はより実践的なソフトウェア開発技術を学習することが期待できる．

しかし，PBL は企業が行うソフトウェア開発プロジェクトとは異なり，プロジェクトメンバはソフトウェア開発を学習しながら，ソフトウェア開発に参加する．そのため，十分なソフトウェア開発技術を持たないままソフトウェア開発を行っている可能性が高い．それに伴い，ソフトウェア開発で発生し得るリスクが顕在化しやすいと考えられる．この顕在化しやすいと考えられるリスクのうちの一つに見積りの甘さによる納期遅れが挙げられる．ソフトウェア開発における見積りの手法としてファンクションポイント法 (FP 法) を代表とする規模見積もり，や COCOMO(constructive cost model) などの工数見積り手法があるが，工数を見積るためにはまず規模見積もりを行わなければならない．しかし，規模見積もり手法である FP 法では，ファンクションポイントを算出するためには，開発するソフトウェアのデータファンクションとトランザクショナルファンクションの計測が必要である．この二つのファンクションポイ

Development of a Man-hour Estimation Tool which Uses the Past PBL Data.

Takeru Saito　Ayahiko Niimi　Kei Ito, 公立はこだて未来大学, Future University Hakodate.

ントは計算者によってその分類の仕方が変わるため，学習者が安定して計測するまで慣れと経験が必要である．

また，ソフトウェア開発 PBL を行っている大学等の教育機関では，開発したソフトウェア以外にも，要件定義書や仕様書等のドキュメントや会議毎に書かれた議事録等，様々なデータが過去に行われた PBL で蓄積されている．開発の際に Redmine などのプロジェクト管理ツールや Subversion などのバージョン管理システムを利用していた場合，参加者ごとの詳細な活動履歴を得ることができる．だが，現在これらのデータは蓄積されているばかりで十分に利用されていない．

そこで本研究では，これらの十分に活用されていない過去の PBL から得られるデータを利用した，工数見積りツールの開発を行う．

2 関連研究

2.1 PBL の支援環境開発

ソフトウェア開発 PBL を支援する試みとして，福安ら [1] は作業リポジトリのガントチャート出力を利用したグループ並行型 PBL を提案している．この研究では，PBL 活動日での活動には Web アプリケーションによるガントチャート表示，非活動日には活発に活動が行われている時間帯を中心にガントチャート表示を行った．また，明石ら [2] はレビューと改善作業に重点をおいた PBL のコース管理システム Collasys を開発した．Collasys は成果物のレビュー項目ごとに掲示板を作成し，この掲示板上で作成担当者とレビュアーが修正方針に関して議論しながら，修正を行うことができる．これらの PBL 支援システムでは，プロセスモニタリングや相互評価によって PBL の支援を行っているが，プロジェクトの見積りという観点からの支援システムは確認されていない．

2.2 工数見積り手法

ソフトウェア開発の工数見積りを行う際，まず対象となるプロジェクトの規模の見積りを行う．これは，開発規模と開発工数が比例していると言う仮定を元にしているためである．このソフトウェア開発の規模を見積る手法として，FP 法が一般的に利用されている．FP 法は，対象となるソフトウェアプロジェクトの入力，出力，照会，内部論理ファイル、外部インターフェースファイルの 5 つの外部属性に注目し，計測したファンクションポイントをそれぞれの複雑度ごとに重み付けを行う [3]．このファンクションポイントの集計を利用することで，ソースコードの規模だけでなく，文書やバグの規模見積もりも可能となる．

しかし，ソフトウェア開発プロジェクトには，開発環境やプロジェクトメンバーの習熟度などが生産性を変動させる要因となるため，開発工数の見積りの際にはファンクションポイントだけでは見積りが難しい．そのため開発工数を導出するためには，ソフトウェアの規模だけでなく，それらの生産性要因を説明変数としなければならない．この説明変数に利用する生産性要因を決定する方法は，相関係数を利用する方法などがあるが，天嵜 [4] は 5 つの変数選択手法の比較を行っている．その結果として，対象となるデータサイズが極端に小さくない限り，ステップワイズ法が最も妥当であることが示された．また，生産性要因が定性的または定量的に記録されていない場合，その生産性要因を見積りに利用できない．角田ら [5] はこの問題に対し，プロジェクトマネージャが生産性要因の生産性レベルを荒い粒度で推測し，それを説明変数とすることを提案している．

そして，これらの説明変数を利用する工数の見積り手法は大きく分けて以下の 6 つがある [6]．

- 類推法
- 専門家による判断
- トップダウン見積り
- ボトムアップ見積り
- パラメトリック法
- プライスツーウィン

類推法とは見積り対象プロジェクトと類似した過去のプロジェクトの実績を基に見積りを行う手法である．まず全体の工数を見積ってから，各作業に分配するのがトップダウン見積り，逆に成果物の構成要素ごとに工数を見積り，その結果を積み上げるのがボトムアップ見積りである．パラメトリック法は工数を目的変数とし，プロジェクトの工数の変動要因とな

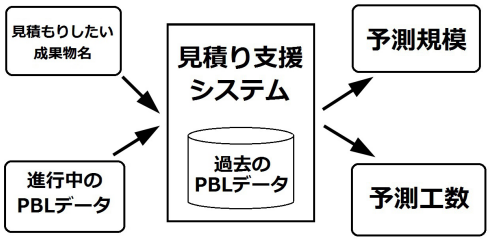

図 1　システムの入出力

る要素を説明変数と設定し，数学的な手法で工数を見積る手法である．パラメトリック法に属する代表的な見積り手法に，COCOMO や CoBRA 法がある [6]．COCOMO は回帰分析を利用したプログラム言語の難度に左右されない手法で，COBRA 法は熟練者の経験とソフトウェア開発データを利用し工数を見積る手法である．また，プライスツーウィンは顧客予算を基に見積りを行う手法だが，見積もり手法としてはふさわしくないとされている．

3　アプローチ

3.1　見積りの流れ

本研究では，過去に実施されたソフトウェア開発 PBL で蓄積されたデータより，開発環境やプロジェクトメンバの習熟度などの目的変数を抽出し，見積もりのためのデータセットとして利用する．ここで，本研究で制作する見積り支援システムの入出力構成を図 1 に示す．図 1 の通り本システムでは，進行中のソフトウェア開発 PBL のデータと見積りしたい成果物名を入力する．そしてそれらのデータと，システム内に蓄積した過去に実施されたソフトウェア開発 PBL のデータから予測規模と予測工数を割り出し，出力する．

3.2　利用データ

本研究では，過去の PBL で蓄積したデータを，バージョン管理システムである Subversion とプロジェクト管理システムの Redmine から取得する．

3.2.1　Subversion

Subversion とは，ソフトウェアのソースコードなどをのファイルの管理を行う集中型バージョン管理システムである．このシステムはサーバ上にバージョン管理を行うファイルを格納するリポジトリを用意することで，複数人が最新バージョンをサーバからコピーすることができ，前バージョンへの復元も容易にできる．本研究では，この Subversion のコミットログを利用データの一つとする．

この Subverison のコミットログの中から，以下の 6 種類のデータが獲得できる．

- リビジョン番号
- コミットしたユーザ名
- コミット日時
- 該当リビジョンで追加や変更があったパス
- 変更の種類
- コミットメッセージ

リビジョン番号とはコミット毎に増加する，ソースコードやドキュメントの細かい変更を表す改訂番号のことである．また，該当リビジョンで追加や変更があったパスが，そのリビジョンにおける追加および編集を行った成果物と等しい．一度のコミットで複数のファイルやディレクトリが変更されることがあるため，この変更があったパスはコミット一件につき複数個取得できる．変更の種類というデータは，それぞれのパスが追加，変更或いは削除のいずれかの操作を受けたかを AMDR のアルファベット一文字で表現したものである．コミットメッセージとは，コミット者がコミットを行う際に付加する，作業内容の説明文章である．このコミットメッセージだけは，コミット者が任意で記述を行うため，他のデータと異なり欠損が存在する．

3.2.2　Redmine

Redmine とは，タスク管理や進捗管理が行えるプロジェクト管理システムの一つである．Redmine はチケットと呼ばれる各タスクのデータを登録，編集することで，タスクの進捗管理を行う．また，文章の保存や掲示板の作成ができるため，PBL ではこの機能を利用して会議の議事録を保存，共有している．そこで本研究では，Redmine に蓄積された過去のプロ

ジェクト情報を，登録チケットおよび議事録から抽出をする．

Redmine のチケットから獲得できるデータは以下の 16 種類である．

- チケット番号
- プロジェクト名
- トラッカー (チケットの種類)
- チケット名
- 親チケット名
- ステータス
- 優先度
- 作成者
- 担当者
- 作成日
- 更新日
- 開始日
- 期日
- 対象バージョン
- 進捗％
- 作業時間の記録

このデータの中で，トラッカーとはチケットの作業内容の分類を示し，バグ，機能，サポート，タスクのいずれかが格納される．また，チケットには既に作成されたチケットを細分化して，詳細なタスクとして新たにチケット登録をすることがある．そのとき細分化したチケットに，元となったチケットを親チケットとして登録をする．チケットの進捗状況を表す要素として，ステータスがある．ステータスには着手，進行中，完了などが入力される．対象バージョンとは，そのチケットによる修正がどのバージョンに取り込まれるかが入力されたデータである．しかし，管理者が該当するバージョンを Redmine に設定していない場合，このデータは欠損する．

また議事録からは，以下のデータが収集し，利用する．

- 会議の日時
- 会議場所
- 参加メンバ
- 会議内容
- 次回の活動予定
- 作成担当者

3.3 見積り手法

PBL では一般的にメンバの増員を想定していないため，納期までに作業を間に合わせるためには，開発するシステムのスコープかドキュメント等の成果物の作成予定を調整するのが一般的である．そのため，各成果物ごとの開発工数を見積ることができれば，成果物作成予定の調整に役立つと考えられる．よって本研究では，ボトムアップ見積りによって各成果物ごとの開発工数を見積るのシステムを開発する．

4 収集データ

本研究では，以下のようなデータを情報系大学のソフトウェア開発講義で過去に行われた PBL から入手する事ができた．

4.1 収集できたプロジェクト

本研究において A，B，C，D，E，F，G，H，I，J，K の 11 プロジェクトからデータを収集することができた．

これらプロジェクトの開発期間とメンバ数，および参加学年とそのプロジェクトの開発種別を表 1 に示す．この 11 プロジェクトのうち，A から E までの 5 プロジェクトが学部 3 年生，F から K までの 6 プロジェクトが大学院 1，2 年生が参加した．表 1 の開発種別とは，開発するソフトウェアが新規に開発されたものなのか，既存のものを拡張あるいは再開発したものなのかを示しており，プロジェクト D，J 以外は全て新規開発である．また，F から I までのプロジェクトの開発期間が 0.16ヶ月となっているが，これは 5 日間のみの活動であったことを示している．

4.2 Subversion のコミットログ

Subversion のログデータは，全てのプロジェクトデータから取得できた． 各プロジェクトの取得リビジョン情報を表 2 に示す．A，B，C，D，E の 5 プロジェクトの Subversion リポジトリは，主にソースコード以外の UML 図や要件定義書などのドキュメントを管理している．ただし，D，E に関しては，ド

表 1　データ収集したプロジェクト一覧

プロジェクト	A	B	C	D	E	F	G	H	I	J	K
開発期間 (月)	2	2	2	8	8	0.16	0.16	0.16	0.16	4	4
メンバ数 (人)	5	5	5	5	10	5	5	5	5	7	8
参加学年	3	3	3	3	M1,2	M1,2	M1,2	M1,2	M1,2	M1,2	M1,2
開発種別	新規	新規	新規	再開発	新規	新規	新規	新規	新規	機能拡張	新規

表 2　プロジェクトごとの取得リビジョン情報

プロジェクト	A	B	C	D	E	F	G	H	I	J	K
取得リビジョン数	64 件	27 件	69 件	479 件	467 件	23 件	45 件	327 件	109 件	47 件	196 件
一月あたりの リビジョン数	32 件	13.5 件	34.5 件	59.9 件	58.4 件	143.8 件	262.5 件	2043.8 件	681.3 件	11.8 件	49 件
ソースコード 有無	なし	なし	なし	あり	あり	あり	あり	あり	あり	あり	あり
ドキュメント 有無	あり	あり	あり	あり	あり	なし	なし	なし	なし	あり	あり
コミットメッセージ欠損率	65.6%	48.2%	10.1%	54.7%	9%	56.5%	64.4%	82.6%	91.7%	70.2%	35.7%

キュメントだけでなく開発したシステムのソースコードを管理している Subversion フォルダから，ソースコードのリビジョン情報を取得できた．また，F，G，H，I，J，K の 6 プロジェクトはほぼソースコードのみを管理していたため，ソースコードのリビジョン情報を Subversion からデータを取得した．

表 3 はプロジェクト D の報告書作成開始時と作成終了時のログデータである．この表を見ると，作成開始時点では報告書作成のためのディレクトリや編集するファイルの作成を行っており，終了時には報告書を PDF ファイルとして生成していることが分かる．

これらの Subversion のコミットログからは，各成果物の作成を行った人物と，初回コミットから最終コミットまでの期間を抽出することができる．それにより，過去の PBL から各成果物の作成に必要となった人日を推定することができると考えられる．特に，プロジェクト A，B，C，D，E からはソースコード以外のドキュメントの作成工数が抽出できる．

4.3　Redmine のチケット

11 プロジェクトそれぞれから得られた Redmine のチケット総数と開発期間あたりのチケット数を表 4 に示す．表 4 より，チケット総数は開発期間とやや比例しているが，プロジェクトによって発行数の差が大きい．また，開発期間の短い F から I までのプロジェクトは短期間で集中してチケット発行を行ったため，開発期間あたりのチケット数が増加したと考えられる．

表 5 はプロジェクト D のグループ報告書作成を示すチケットデータである．この表を見ると，報告書作成が急いで行わなければならなく，また開始日と期日から 10 日間を必要な作成期間であると見積ったことがわかる．

この Redmine のチケットからは，実際に行われた具体的なアクティビティとその担当者のデータが獲得できる．また，そしてプロジェクト中での対象アクティビティの重要度も収集できるため，この PBL がどのような点に気を配っていたかを知ることができる．それによって，対象プロジェクトの特性が判断できると考えられる．Redmine ではチケットに記したタスクの開始日と期日から，その PBL での見積り工

表 3　グループ報告書の Subversion ログデータの例

リビジョン番号	r1175	r1891
コミットしたユーザ名	a	b
コミット日時	2013-12-17 13:18:32 +0900 (2013 年 12 月 17 日 (火))	2014-01-15 16:32:54 +0900 (2014 年 01 月 15 日 (水))
該当リビジョンで追加や変更があったパス	A/final_report/D_Group A /final_report/D_Group /midterm_group_ver0.2.tex 他	M /final_report/D_Group /midterm/8.tex M /final_report/D_Group /midterm_group_ver0.2.pdf 他
変更の種類		
コミットメッセージ	最終報告書のテンプレート (中間報告書のコピー) をコミットしました。	

表 4　プロジェクトごとの取得チケット情報

プロジェクト	A	B	C	D	E	F	G	H	I	J	K
取得チケット数	25 件	17 件	25 件	160 件	240 件	48 件	22 件	10 件	45 件	4 件	44 件
一月あたりのチケット数	5 件	3.4 件	5 件	20 件	30 件	300 件	137.5 件	62.5 件	281.3 件	1 件	11 件

表 5　グループ報告書のチケットデータの例

チケット番号	1067
プロジェクト名	D
トラッカー	サポート
チケット名	グループ報告書の作成を行う
親チケット名	最終報告書の作成を行う
ステータス	完了
優先度	急いで
作成者	c
担当者	c
作成日	2013/12/18 18:49
更新日	2014/01/20 13:35
開始日	2013/12/18
期日	2013/12/28
対象バージョン	D 班
進捗%	100
作業時間の記録	0

数も抽出できる．しかし，更新日が同一のチケットが複数あることから，すでに完了した複数のタスクを一括して更新していることが窺える．そのため，開始日と更新日の差だけから，実際に掛かった工数を抽出するのは難しいと考えられる．実際の工数を抽出する場合には，チケットの開始日と前述の SVN の最終コミットの日付から割り出す必要がある．

4.4　議事録

議事録は，Redmine の文書機能やフォーラム機能を用いて記録されたものが取得できた．表 6 より，議事録は開発期間が長いほど，Redmine に登録されている議事録数が多くなっていることが分かる．プロジェクト F と H からは議事録を収集できなかったが，その他のプロジェクトでは定期的に議事録が作成されていると考えられる．

議事録から得られたデータのうち，日時と場所，そして参加メンバの情報はどの議事録でも，まず間違いなく獲得できた．しかし，次回の活動予定と作成担当者に関するデータは欠損が見られた．議事録のフォーマットは各プロジェクトごとに定義されており，学内で統一されたフォーマットがないため，作成担当者や活動予定を記入しないプロジェクトもあることが分かった．また，会議内容はプロジェクトの進捗状況により記述される内容が異なり，特に要件定義や設計の

表 6　プロジェクトごとの取得議事録数

プロジェクト	A	B	C	D	E	F	G	H	I	J	K
取得議事録数	13 件	13 件	13 件	50 件	61 件	0 件	2 件	0 件	2 件	5 件	24 件
一月あたりの議事録数	6.5 件	6.5 件	6.5 件	6.3 件	7.6 件	0 件	12.5 件	0 件	12.5 件	1.3 件	6 件

段階ではパラメトリック法で利用できるメトリクスを獲得できた．たが，議事録ごとに大幅に粒度が異なっており，会議の際に上がった意見等を詳細に書き記したものから，当日に行った活動のみを記述するのみにとどまる場合もあった．

表 7 はプロジェクト D がグループ報告書を作成している期間中の議事録の一部である．この表からは，詳細な活動の内容と報告書の作成担当にどのメンバが割り当たっているのかが分かる．

この議事録から得られるデータには，PBL の開発体制や作成ソフトウェアが新規開発か否か，報告会など納期の目安となる日程など，PBL の対象となるソフトウェア開発プロジェクトの基本的な性質を示すものが多い．そのため，これらのデータは見積りを行うに当たって，全ての成果物に関わる基本的なメトリクスとして利用できる可能性がある．

5　見積り実験

4 章で得られたデータより，プロジェクトの規模と工数の見積りを行う．工数見積りを行う上で必要とされる，開発環境等の工数の変動要因の多くは議事録から得られることが分かった．また，Subversion のログと Redmine のチケットを照らし合わせることで，予測工数と実際の工数を獲得できる．この過去に行った成果物作成の工数と，工数の変動要因を元に各成果物の作成工数を目的変数とする数学的な関数を導きだせると考えられる．類似法を用いて成果物作成の見積りを行うことも可能であるが，これを行うためにはより多くの PBL データが必要となる．現状では類似法による成果物作成の工数見積りが十分に行えるほどの PBL データが揃っていない．そのため，本研究では一成果物の工数見積りは類似法ではなく，パラメトリック法による工数見積り関数の作成と利用が妥当であると思われる．

4.2 節と 4.3 節，および 4.4 節で得たデータより，各成果物の推定作業工数を表 8 に表す．表 8 は各成果物の作成工数を人日で表している．表中に小数点以下の人日を取る工数は，作業が 1 日の間に完了したものを示している．また該当成果物を作成していなかったり，作業期間のデータが欠損しているものは N/A と表記している．この表 8 と表 1 を照らし合わせると，プロジェクト全体の開発期間とソースコードの作成工数は必ずしも比例しないことが分かる．これは，ソースコード以外の成果物をどれだけ作るかによって，開発期間をコーディング作業が占める割合が変わるためであると考えられる．

また，表 8 の画面遷移図に関して，プロジェクト B では 69 人日で開発されたが，同じ開発期間で実施されたプロジェクト A および C ではそれぞれ 10 人日，5 人日と計測できた．ここで，表 2 と表 4 を見ると，プロジェクト B はプロジェクト A，C に比べて取得リビジョン数，取得チケット数が少ないことが分かる．そのため，画面遷移図作成の際には Subversion のコミット頻度と Redmine のチケット登録数が工数に影響する可能性があることが示唆された．

6　提案システムの構成

5 節より，本システムでは以下のような処理によって工数見積りを実現する．

1. システム利用者から進行中のプロジェクトのメンバ数や開発種別などのプロジェクトの基本情報と，見積り対象とする成果物名の入力を受ける．
2. システム利用者から Subversion のコミットログおよび Redmine のチケット情報を受け取る．
3. 入力情報を元に開発規模の見積り値を算出する．
4. 過去の PBL データを元に見積り対象成果物の

表 7　グループ報告書に関する議事録の例

会議の日時	2013/12/18 4,5 限
会議場所	484 室
参加メンバ	b，c，d，e，f
会議内容	グループ別作業 報告書班 (c，d，e):イベント版図の意見あわせ, 目次, 担当割り決定 (担当割りはレポジトリの表で確認できる)
次回の活動予定	報告書班:金曜日 2 限に集めて進捗具合確認と今後の予定確認
作成担当者	d

表 8　各プロジェクトにおける成果物ごとの推定実績工数 (人日)

成果物	要件定義書	ユースケース記述	画面遷移図	ER 図	ソースコード	発表用スライド	報告書
A	89	55	10	0.09	215	133	210
B	70	63	69	28	140	133	210
C	64	44	5	72	140	133	210
D	155	112	72	90	410	63	125
E	N/A	N/A	10	3	1660	126	370
F	N/A	N/A	N/A	0.04	10	0.17	N/A
G	N/A	N/A	0.01	N/A	10	1	N/A
H	N/A	N/A	N/A	N/A	10	1	N/A
I	N/A	N/A	0.16	N/A	8	1	N/A
J	49	N/A	N/A	N/A	392	N/A	N/A
K	14	N/A	N/A	N/A	704	N/A	N/A

調整係数を作成する．

5. 利用する工数見積り関数にシステム利用者から取得したデータと調整係数を代入し，対象成果物の見積り工数を算出する．
6. 算出した見積り規模と見積り規模を出力する．

仮に工数見積り手法として CoBRA 法を利用する場合，手順 5 では以下の式を利用する．

$$Effort = \alpha Size(1 + \sum CO_i) \qquad (1)$$

この式 1 の *Effort* が工数を表し，*Size* が見積り規模を表す．ここで，式 1 の α および CO_i が手順 4 で算出する調整係数となる．

7　まとめと今後の課題

本研究ではソフトウェア開発 PBL を支援するため，過去の PBL で蓄積されたデータを利用した成果物ごとの工数見積りツールを提案する．このツール制作のため，現在取得できるデータの検証を行った．その結果，11 のソフトウェア開発 PBL から 7 種類の開発成果物の開発工数を抽出することができた．

今後は取得できたデータを元に，各成果物に対する適切な規模および工数見積り法の検討が必要である．特にソフトウェア開発 PBL ではプロジェクトメンバごとに開発スキルが大きく異なる可能性があるため，その差異にも対応できる必要がある．この個人の開発スキルに関するデータが取得可能か否か，入手できな

い場合にメンバの学年が利用できるかどうかの検討も必要となる.

参 考 文 献

[1] 福安直樹, 佐伯幸郎, 水谷泰治：リポジトリのリアルタイムな可視化にもとづく PBL の支援環境： 継続的な実施を目的として, 電子情報通信学会技術研究報告. SS, ソフトウェアサイエンス, Vol. 110, 458(2011), pp. 1–6.

[2] 明石敬, 松澤芳昭, 大岩元：Project-Based Learning を支援するコース管理システム, 情報処理学会研究報告. コンピュータと教育研究会報告, Vol. 2007, No, 12(2007), pp. 15–22.

[3] Copers Jones(1998)：Estimating Software Costs, McGraw-Hill Osborne Media, (富野壽 (訳) (2001), ソフトウェア見積りのすべて：規模・品質・工数・後期の予測法, 共立出版).

[4] 天嵜聡介：工数見積もり手法における変数選択手法の比較, 情報処理学会研究報告. ソフトウェア工学研究会報告, Vol. 2006, No. 125(2006), pp. 1–8.

[5] 角田雅照, 門田暁人, ジャッキー キョン, 松本健一：熟練者判断を取り入れたソフトウェア開発工数見積りモデル, 情報処理学会論文誌, Vol. 55, No. 2(2014), pp. 994–1004.

[6] IPA 独立行政法人 情報処理推進機構：エンタプライズ系事業/見積もり手法, http://www.ipa.go.jp/sec/softwareengineering/std/ent01-c.html, (2014/07/15 アクセス).